Compositional Modular Scheme

Guruduth Banavar*       Gary Lindstrom
Department of Computer Science
University of Utah, Salt Lake City, UT 84112

Abstract. We present a new module system for Scheme that supports a high degree of implementation reuse via module composition. The module system encourages breaking down a program into the smallest possible individually meaningful modules, and recomposing them using a powerful set of adaptation and combination mechanisms. Even hierarchical nesting is achieved via a composition operation. This module system is shown to support a stronger and more flexible notion of compositionality and reuse than traditional class-based inheritance in object-oriented programming. Finally, this module system is itself implemented by reusing a language independent OO framework.

Keywords: module systems, object-oriented programming, inheritance, Scheme.

1 Introduction

Modularity is a fundamental facility for controlling complexity in large systems, via decomposition and abstraction. In particular, software modules allow programmers to develop and maintain pieces of a large system relatively independent of each other. However, decomposition alone does not support reuse of software components, which is widely accepted to aid the efficient construction of large systems. For this, it is necessary to provide mechanisms for effective recomposition, by which conforming modules can be composed to obtain other modules.

Compositional modularity is a model that supports a simple notion of modules along with a powerful notion of their composition. In addition to meeting requirements of large-scale software development such as encapsulation, separate development, and checking of inter-module conformability, the distinguishing goal of this model is to enable maximal reuse of software components. It encourages breaking down software into the smallest possible individually meaningful units, then recomposing them in various ways to get larger modules. Aspects of modules can be adapted in several ways to make them suitable for composition in new ways.

In essence, compositional modularity distills, unifies, and further advances many existing notions of modularity. In particular, this includes varieties of class-based OO programming, in which inheritance is the primary mechanism for implementation composition to create new classes (modules). Traditionally, OO inheritance is a composite notion, involving module extension, attribute rebounding, encapsulation, static binding, etc. In contrast, individual aspects of inheritance are achieved in our model using separate operations, which can be used in combination to emulate

* Contact: e-mail: banavar@watson.ibm.com, Phone: +1-914-784-7755, fax: +1-914-784-7455.